

[First Hit](#) [Fwd Refs](#)[Previous Doc](#) [Next Doc](#) [Go to Doc#](#)☐ [Generate Collection](#) [Print](#)

L11: Entry 5 of 15

File: USPT

Jul 9, 2002

DOCUMENT-IDENTIFIER: US 6418421 B1

TITLE: Multimedia player for an electronic content delivery system

Detailed Description Text (170):

A one-way hash algorithm is used to calculate a message digest. A hash algorithm takes a variable-length-input message and converts it into a fixed length string, the message digest. A one-way hash algorithm operates only in one direction. That is, it is easy to calculate the digest for an input message, but it is very difficult (computationally infeasible) to generate the input message from its digest. Because of the properties of the one-way hash functions, one can think of a message digest as a fingerprint of the message.

Detailed Description Text (215):

122 Metadata is captured from the Content Provider(s)' Database 160 by the Content Information Processing Subsystem using the Content Provider(s)' 101 unique identifier for the Content 113 and information provided by the Database Mapping Template.

Detailed Description Text (258):

The Rights Management Architectural Model is shown in FIG. 5 and this illustrates the mapping of the architectural layers to the operating components making up the Secure Digital Content Electronic Distribution System 100 and the key functions in each layer.

Detailed Description Text (319):

Electronic Digital Content Store(s) 103 download the Metadata SC(s) 620, for which they are authorized, and build Offer SC(s) 641. In short, an Offer SC(s) 641 consists of some of the parts and the BOM from the Metadata SC(s) 620 along with additional information included by the Electronic Digital Content Store(s) 103. A new BOM for the Offer SC(s) 641 is created when the Offer SC(s) 641 is built. Electronic Digital Content Store(s) 103 also use the Metadata SC(s) 620 by extracting metadata information from them to build HTML pages on their web sites that present descriptions of Content 113 to End-User(s), usually so they can purchase the Content 113.

Detailed Description Text (328):

The Key Description Part columns of the following table define the records that are included in the Key Description part of the SC(s). Records in the Key Description part define information about the encryption keys and algorithms that were used to encrypt parts within the SC(s) or parts within another SC(s). Each record includes the encrypted part name and, if necessary, a URL that points to another SC(s) that includes the encrypted part. The Result Name column defines the name that is assigned to the part after it is decrypted. The Encrypt Alg column defines the encryption algorithm that was used to encrypt the part. The Key Id/Enc Key column defines either an identification of the encryption key that was used to encrypt the part or a base64 encoding of the encrypted Symmetric Key 623 bit string that was used to encrypt the part. The Sym Key Alg column is an optional parameter that defines the encryption algorithm that was used to encrypt the Symmetric Key 623 when the previous column is an encrypted Symmetric Key 623. The Sym Key ID column is an identification of the encryption key that was used to encrypt the Symmetric Key 623 when the Key Id/Enc Key column is an encrypted Symmetric Key 623.

Detailed Description Text (336):

Watermarking Instructions--A part that contains the encrypted instructions and parameters for implementing watermarking in the Content 113. The watermarking instructions may be modified by

the Clearinghouse(s) 105 and returned back to the End-User Device(s) 109 within the License SC (s) 660. There is a record in the Key Description part that defines the encryption algorithm that was used to encrypt the watermarking instructions, the output part name to use when the watermarking instructions are decrypted, a base64 encoding of the encrypted Symmetric Key 623 bit string that is was used to encrypt the watermarking instructions, the encryption algorithm that was used to encrypt the Symmetric Key 623, and the identification of the public key that is required to decrypt the Symmetric Key 623.

Detailed Description Text (374):

The following table shows the parts that are included in the Order SC(s) 650 as well as its BOM and Key Description parts. These parts either provide information to the Clearinghouse(s) 105 for decryption and verification purposes or is validated by the Clearinghouse(s) 105. The parts and BOM from the Offer SC(s) 641 are also included in the Order SC(s) 650. The Some string in the Part Exists column of the Metadata SC(s) BOM indicates that the some of those parts are not included in the Order SC(s) 650. The BOM from the Metadata SC(s) 620 is also included without any change so that the Clearinghouse(s) 105 can validate the integrity of the Metadata SC(s) 620 and its parts.

Detailed Description Text (411):

The V property specifies the version of the SC(s). This is the version number of the SC(s) specification that the SC(s) was created under. The string that follows should be of the form major.minor.fix, where major, minor, and fix are the major release number, minor release number, and fix level, respectively.

Detailed Description Text (460):

S key_identifier signature_string signature_algorithm

Detailed Description Text (461):

The S record contains the key_identifier to indicate the encryption key of the signature, the signature_string, which is the base64 encoding of the digital signature bitstring, and the signature algorithm that was used to encrypt the digest to create the digital signature.

Detailed Description Text (467):

The key_encryption_algorithm_identifier indicates the encryption algorithm that was used to encrypt the Symmetric Key 623. The encrypted symmetric key is a base64 encoding of the encrypted Symmetric Key 623 bit string that was used to encrypt the part.

Detailed Description Text (581):

If adequate information is provided to enable an automated query to the Database 160 of the Content Provider(s)' 101, the job is queued for Automatic Metadata Acquisition Process 803. If the database mapping table has not been configured for the Automatic Metadata Acquisition Process 803, the job is queued for Manual Metadata Entry Process 804 (see Automatic Metadata Acquisition Process 803 section for details on the Database Mapping Table).

Detailed Description Text (586):

database mapping table with adequate information to generate queries to the Database 160 of the Content Provider(s) 101

Detailed Description Text (741):

To extract the template data fields from the Database 160 of the Content Provider(s) 101 the Automatic Metadata Acquisition Tool uses a table that maps the type of data (e.g., composer, producer, a biography of the artist) to the location within the database where the data can be found. Each of the Content Provider(s) 101 help specify that mapping table for their environment.

Detailed Description Text (742):

The Automatic Metadata Acquisition Tool uses a metadata template of the Content Provider(s) 101 and mapping table to acquire whatever data is available from the Databases 160 of the Content Provider(s) 101. The status of each product is updated with the result of the Automatic Metadata Acquisition Process 803. A product which is missing any required data is queued for

Manual Metadata Entry Process 804, otherwise it is available for packing into a Metadata SC(s) 620.

Detailed Description Text (883):

Metadata SC(s) 620 received into a new content directory via FTP from the Content Dispersement Tool is processed by the Content Promotions Web Site 156. These containers can be opened with the SC(s) Preview Tool to display or extract information from the container. This information can then be used to update HTML Web pages and/or add information to a searchable database maintained by this service. The SC(s) Preview Tool is actually a subset of the Content Acquisition Tool used by the Electronic Digital Content Store(s) 103 to open and process Metadata SC(s) 620. See the Content Acquisition Tool section for more details. The Metadata SC(s) 620 file should then be moved to a permanent directory maintained by the Content Promotions Web Site 156.

Detailed Description Text (884):

Once the Metadata SC(s) 620 has been integrated into the Content Promotions Web Site 156, its availability is publicized. The Content Provider(s) 101 can send a notification to all subscribing Electronic Digital Content Store(s) 103 as each new Metadata SC(s) 620 is added to the site or can perform a single notification daily (or any defined periodicity) of all Metadata SC(s) 620 added that day (or period). This notification is performed via a standard HTTP exchange with the Electronic Digital Content Store(s) 103 Web Server by sending a defined CGI string containing parameters referencing the Metadata SC(s) 620 added. This message is handled by the Notification Interface Module of the Electronic Digital Content Store(s) 103 which is described later.

[Previous Doc](#)

[Next Doc](#)

[Go to Doc#](#)

[First Hit](#) [Fwd Refs](#)[Previous Doc](#)[Next Doc](#)[Go to Doc#](#)

Generate Collection

Print

L11: Entry 13 of 15

File: USPT

May 8, 2001

DOCUMENT-IDENTIFIER: US 6230168 B1

TITLE: Method for automatically constructing contexts in a hypertext collection

Abstract Text (1):

A method of facilitating hypertext navigation within a collection of linked files includes gathering a set of linked files for inclusion within the collection and identifying a map of links among the files. The map is then analyzed to determine groupings among the files based on relevance. Additional hypertext links are then created that link to files within the at least one group. The additional hypertext links are then stored in one of the files in the collection a new file that is added to the collection.

Brief Summary Text (9):

The method includes gathering a set of linked files for inclusion within a collection and identifying a map of links among the files. The map is then analyzed to determine groupings among the files based on relevance. Additional hypertext links are then created that link to files within the at least one group. The additional hypertext links are then stored in one of the files in the collection a new file that is added to the collection.

Drawing Description Text (12):

FIG. 10 depicts the mapping of token ranges from two predetermined dictionary files into a stored dictionary file according to a preferred embodiment of the present invention.

Detailed Description Text (4):

Each of the servers 12 has a unique identifier or name that can be used to communicate with it. Typically these server names are expressed as character strings, for example www.ibm.com that are translated by mechanisms in the network to unique numeric addresses pursuant to the well known internet Domain Name System ("DNS"). The server's name along with additional characters that identify a specific web-page constitute an identifier known as a Uniform Resource Locator (URL). While this information can be anything that identifies the requested material, this additional information is frequently file and/or directory names that reference the file system on the server 12.

Detailed Description Text (5):

A common method of accessing information on the internet is with a browser residing on the client computer 10. Toward this end, information on the server computers 12 is typically stored in a format that is usable by browsers. Typically, this format is hypertext markup language ("HTML"), which is readable. The browser enables a user at the client computer 10 to receive HTML files transmitted from server computers 12 using widely available protocols, including the hypertext transfer protocol ("HTTP"). Once received, the HTML file is rendered by the browser as a page (or "web page"). Rendering includes displaying text, pictures, and video and playing recorded sound. Commonly available browsers include Netscape Navigator, Mosaic, and Internet Explorer supplied by Microsoft corporation.

Detailed Description Text (9):

The most powerful feature of HTML is the hypertext link tag. This tag allows an author of one HTML file to specify links to a plurality of other HTML files. Thus, when an HTML file is displayed as a web page on a user's browser, the user may select one of the hypertext links and instantly redirect their browser to request and render the web page corresponding to the selected link.

Detailed Description Text (12):

The methods used to determine the scope of hypertext material, such as but not limited to tagged HTML, SGML or XML files, to be included in a collection may either be manually specified or automatically developed from the linked structure of the material. Manual specification identifies existing sets of files, database entries, or previously existing collections to be included in or excluded from the collection process. For example, on the World Wide Web, such manual specification might include lists of server sites, file directories or URL text string patterns that are to be considered within (or ruled out of) the scope of the binding operation.

Detailed Description Text (17):

A method for organizing a collection is depicted in FIG. 4. In step 600, the scope of the files included within the collection is determined as described above. In step 602, a graph of the material within the collection is identified as described above. Then, in step 604, the organization of the material within the collection is analyzed and in step 606 organizational aids are added to the collection by an additional process called HyperBinding. HyperBinding is a process that examines the map of the collection to create new pages and links to include in the HyperBound collection to enhance its usability.

Detailed Description Text (18):

The most basic form of this process, called "static hyperbinding", simply uses the information collected in the map constructed by previous steps to construct new links and navigational information such as buttons and menus to allow the user to easily traverse the relationships represented in the collected map. This mechanism creates lists and tables corresponding to the hierarchy of links represented in the map but factors those links away from the body of text. These lists then allow the user to see the structure of the material separately from the body of the data. This works well in many cases because hypertext authors frequently include sufficient links in their material that expose the implied organization of the data even though they do not necessarily include all the navigational pages and links that a user may desire.

Detailed Description Text (19):

In some cases, however, there is little or no explicit structure that can be easily discerned from simple examination of the mapped objects. This may be the case either because there are not many links between the objects in the collection, or because there so many relationships expressed in links that is difficult or impossible to discover which links constitute a valuable organization of the material. When this is the case a more sophisticated technique is employed. What is termed "dynamic hyperbinding" examines the record of relationships created by the previously described steps and analyzes them to detect patterns in the "graph" that could offer organizational value.

Detailed Description Text (20):

One form of dynamic hyperbinding involves automatic structural analysis of the link graph to discover clusters of strongly related material. Based on the mapped structure of links, a link density clustering analysis involves determining where many hyperlinks connect a common set of objects. There will frequently be more than one density cluster identified for a given collection. In a given set of n linked hypertext objects there may be as few as $n-1$ links connecting them or as many as $(n \cdot \sup{2} - n) / (n-1)$ in a fully connected mesh. In determining when a cluster exists the ratio of the number of hypertext objects that constitute the candidate group of objects to the number of links that connect them is a measure of the density of their interconnection. Candidate objects for inclusion in possible clusters may be located either by recursively traversing the graph or more efficiently by locating nodes in the graph that are either pointed to by many others nodes or that themselves point to many other nodes. In discovering clusters, both the number of times a particular object is targeted and also where it is targeted from are important. For example, the link density of a cluster containing an object targeted from 50,000 other objects will be low (near $n-1$) and therefore probably not identified as a cluster. On the other hand, if one object is targeted from a hundred pages, and each of those hundred pages is in fact targeted from a single web page, then the link density of this cluster would be higher (equal to $2 \cdot (n-1)$) and it might be identified as cluster. In a candidate list of pages that were all pointing to each other the density would approach $(n \cdot \sup{2} - n) / (n-1)$ and would almost certainly be a cluster.

Detailed Description Text (22):

In another form of dynamic hyperbinding a different clustering analysis is performed. Here not only the map of the relationship between hype objects is examined, but also the actual language content of the pages is exploited. In document context analysis, linguistic and statistical analysis methods including phrase identification, word frequency analysis, and situational ranking is used to determine the content of a hypertext object. Situation ranking categorizes word and phrase location in the document (such as within a title, heading level, or a list-item) along with simple language frequency. More sophisticated language processing such as grammatical structure and usage analysis may also be employed to categorize the hypertext object. Clusters are then determined by relating files within the collection that have similar semantic or linguistic content.

Detailed Description Text (24):

In both static and dynamic hyperbinding, once the scope of the collection has been specified identified, and possible organizational groups have been identified as is shown in FIG. 5, additional organization aids (typically hypertext links) can then either be automatically added to the collection or interactively added with the participation of an author or editor. FIG. 6 shows two such additional navigational aids. The first navigational aid 320 is a hypertext page added to the collection that appears similar to a "table of contents". The page 320 includes hypertext links 325 and 328 pointing to file A and J respectively. File A and J are hierarchically the first two pages in the clusters 320 and 346 identified in FIG. 5. The page 320 was created by the "static hyperbinding" method of traversing the map created in previous steps. A second navigational aid in the form of a page 330 was also added by hyperbinding. Page 320 contains hypertext links 335 and 338 that point to pages C and I that are at the center (point of highest cluster density of the clusters identified in FIG. 10. Page 330 can be thought of as a "subject" map that is like the "back of the book index" in printed material.

Detailed Description Text (42):

The invention may also employ special encoding for numeric strings that occur in the text. Sometimes when documents contain only occasional references to numbers, no special numeric encoding is required. In these cases each unique numeric string is simply assigned a token in the aforementioned supplemental dictionaries and encoded in the normal way that words and strings not found in any of the predetermined dictionaries are handled. However, when documents contain large numbers of unique numbers this mechanism quickly exhausts the supply of tokens available in the supplemental dictionaries. In these cases one of the two special encoding methods described below can be used.

Detailed Description Text (43):

Numbers, either integers which are just strings of digits, or decimal notation which may include signs (positive or negative), decimal points, currency symbols etc . . . can be encoded using a special predetermined "numeric" dictionary. This dictionary includes a token range (usually one-byte) for each numeric digit (0-9) and also tokens for signs, currency symbols, exponential notation etc. When special encoding for numbers is employed some special parsing rules are also required as numeric stings do not normally include imbedded white space such as is common between other tokenized words or strings. For numeric strings this form of encoding does not achieve any real compression but it allows many numeric strings to be included in documents that also have a lot of regular language text without overloading the token addressing ranges. For documents that have extraordinarily large quantities of unique numeric strings (scientific or financial tables) yet another encoding algorithm is employed by the invention. In these cases, the numbers are converted to a machine independent binary format. This format is designed to successfully encode either integers or real numbers with nominally unlimited precision The format specifically includes information as to how many bits of data are used to encode either very small or very large and precise numbers (mantissa and exponent lengths) and to what precision such numbers are carried in their representation. Normally such encoding schemes will achieve better compression of numbers than the previously described digit tokenization. This form of encoding also more readily allows computation and/or comparison of the numbers which in turn may be used to enable relational searches. For example, with this scheme of encoding, a search could find all numbers less than, equal to, or greater than a particular value. Of course when this form of binary encoding is used there may be precision

limits on the manipulation of the numeric values that are related to the computing capabilities (register size) of a particular hardware or software platform, or the limits of the encoding (compression) algorithm implementation.

Detailed Description Text (44):

After selecting the dictionaries, the selected dictionaries are stored in the compiled file 50 in step 208 of FIG. 8. At this stage, each of the dictionaries stored within the compiled file in step 208 include words, and corresponding numbers or tokens, to uniquely identify many of the words within the text to be compressed. In step 210, each word within the text from the original material is replaced with the number or token corresponding to that word in the stored dictionaries. During compression in step 210, the input text stream is parsed into words and strings and compared with the words in the all the predetermined dictionaries selected by the language identification step 208. Words that appear in any of the selected predetermined dictionaries are replaced with appropriate token values.

Detailed Description Text (45):

Also during step 210, unusual words and strings that are not present in the any of the predetermined dictionaries during compression are identified. In step 212, the unusual words or strings are gathered into at least one supplemental dictionary and assigned token values. Special processing for numeric strings is also handled and they are converted to tokenized data as described above. The compressed token stream is written into a memory buffer and finally stored in the file in large blocks.

Detailed Description Text (47):

After storage of the predetermined and supplemental dictionaries in steps 208 and 212 respectively, in step 213 the dictionaries may be further compressed using relatively standard and well known methods including front and back-end string folding and run length encoding. In this case, the compressed predetermined dictionaries are also stored in the file.

Detailed Description Text (53):

FIG. 11 depicts an exploded view of a client computer 10. The client computer 10 includes a processor 70 coupled over a bus 72 to a modem, 74 a display 76, fixed memory 78, and random access memory (M) 80. The bus 72 enables communication between the various components of the client computer 10. The fixed memory 78 contains programs stored on a computer usable medium and capacity to store additional data on the medium. The processor 70 loads programs into the RAM 80 and executes the programs, which include operating systems and applications. Typical operating systems are Microsoft DOS and Windows, Unix, Aix, and OS2. The applications include a browser 82 for accessing the internet using HTTP protocols and rendering web pages on the user's display. There are many widely available browser programs including Netscape Navigator, Mosaic, and Microsoft's Internet Explorer. The applications may also include a proxy 84, described in more detail below, which interfaces with the browser 82, the processor 70, and the modem 74 over the bus 72. Additionally, a cache 86 may be implemented by the RAM 80 (or the fixed memory 78). The cache 86 is operative to temporarily store data as required by the browser 82 and/or the proxy 84 at the direction of the processor 70.

Detailed Description Text (57):

In FIG. 13, a browser program 82 is running on a client computer 10 that is coupled to an internet access provider 14 (either via a modem or a local area network). When the browser 82 is initiated, it typically is set to an HTML file or web page located at a remote server 12 having a predetermined URL stored in the browser 82. From this location, the user may select a hypertext link to jump to another hypertext page. There are many ways to accomplish this including by typing in a new address and selecting a hypertext link. Assume for purposes of illustration that the user selects the URL of a web page of the remote server 12 depicted in FIG. 13.

Detailed Description Text (64):

Even while the browser 82 is rendering a requested web page, the proxy may continue to receive and/or request additional compressed text and objects on other web pages included within the scope of the compiled HTML document. For example the proxy may process the tag tree and prefetch material corresponding to hypertext links within the information being rendered by the

browser as described below.

Detailed Description Text (69):

Decompression of the tokenized material for rendering at a display screen or printing is similarly very efficient. Since all the tokens that represent words or strings have fixed lengths (usually, but not always, 1-4 bytes) and are organized in blocks on the file, a decompression routine can pass rapidly through the token stream and find each token without any real parsing. This also means that, unlike in many schemes that employ end-to-end binary compression, a portion of the file composed only of some data from one or more blocks can be decompressed without reading or decompressing any other block from secondary storage (disk) or memory.

Detailed Description Text (70):

In addition to the reduced time to transmit or retrieve the stored material the compressed tokenized form of the text can be searched directly without requiring decompression. This is accomplished by using the referenced predetermined and supplementary dictionaries to "lookup" the tokens for any search words, phrases or strings that are to be found. Once these search targets are appropriately tokenized with the same vocabulary that was used to encode the source material the required token sequences can be searched for instead of the actual character strings. Since, in general the strings of bits that must be compared in such a tokenized matching are much shorter than the words they represent the search can proceed much more rapidly.

Detailed Description Text (72):

Transmission of information between a remote server 12 and client computers 10 can be optimized by prefetching the hypertext links on a requested web page and transmitting their contents to the client computers 10 that are viewing the web page. When a user takes a link to a web page that has been prefetched, the web page already resides in a cache 84 on the client computer 10 and therefore, the browser 82 only needs to render it rather than to fetch it and render it. This reduces the time required to satisfy and render the browser's request for the linked web page.

Detailed Description Text (73):

Prior art programs can be added to existing web browsers to examine the received hypertext material for links and to make asynchronous requests for the referenced pages. These programs, however, cannot determine which of the links on the origin requested web page the user is most likely to choose.

CLAIMS:

1. A method of automatically adding organizational information to a collection of linked files comprising the steps of:

gathering a plurality of linked files for inclusion within the collection of linked files;

identifying a map of links among the plurality of linked files;

analyzing the map of links to determine a group among the plurality of linked files based on a characteristic common to each file in the plurality of linked files;

creating additional hypertext links that link to files within the group by hyperbinding; and

storing the additional hypertext links in one of the plurality of linked files or in a new file that is added to the collection.

3. The method according to claim 1 wherein the group is determined based on a further step of link density analysis of the map of the collection.

[Previous Doc](#) [Next Doc](#) [Go to Doc#](#)

[First Hit](#) [Fwd Refs](#)[Previous Doc](#)[Next Doc](#)[Go to Doc#](#)

Generate Collection

Print

L13: Entry 5 of 16

File: USPT

Jul 1, 2003

DOCUMENT-IDENTIFIER: US 6585777 B1

TITLE: Method for managing embedded files for a document saved in HTML format

Brief Summary Text (4):

Users have clear expectations of how embedded content management should work. These expectations have been established from years of using traditional desktop productivity tools, such as word processing programs, which typically enable both embedding content in a primary document and editing the embedded content. In contrast, for Hyper Text Mark-up Language (HTML)-formatted documents, such as web pages, each piece of content is required to be a separate linked file. In other words, HTML does not directly support the concept of embedding content in the primary document. Nevertheless, the expectations of users have not changed in this HTML-formatted document environment because they still desire HTML documents to support the characteristics of embedded content.

Brief Summary Text (5):

Referring to FIG. 1, when a user saves an electronic document as a typical word processing file, such as a Microsoft "WORD 97" program file shown in a display 100, both a sunburst image 102 and a background image 104 are physically contained in the file as "embedded" files. In contrast, a linked logo 106 and a hyperlink to another web page 108 remains outside of the file as "linked" items. Users experienced with traditional desktop productivity applications have certain expectations in the characteristics exhibited by embedded content within an electronic document, such as the content presented by the display 100. For example, users typically expect the following representative results, shown in Table I, in response to manipulating a electronic file containing an embedded file or operating directly upon an embedded file.

Brief Summary Text (6):

However, when the electronic document of FIG. 1 is saved as an HTML-formatted web page, the sunburst image 102 and the background image 104 can not be physically embedded within the electronic document because of the inherent limitations of the HTML file format. Although the user may believe that the sunburst and the background images 102 and 104 are embedded images, the act of saving the document as an HTML file results in linking these images as separate files to the document. Consequently, prior HTML-compatible editors fail to satisfy the above-referenced expectations of typical users for the performance of embedded files in electronic documents. By linking files, rather than physically embedding files, as a consequence of the HTML format, a user's editing operations may result in the undesirable problem of multiple "orphaned" files that waste disk space and cause general user confusion.

Brief Summary Text (7):

Although the prior art has attempted to solve the problem of managing embedded content in several different ways, each prior solution suffers from key limitations. One prior solution is to present a dialog in response to conducting an HTML save operation, thereby prompting users to select the names and storage locations of each embedded piece of content, while internally converting this content to linked content. For the example of a web page "Web Page.htm" having three different pasted pictures, upon initiating a save operation, the user is typically presented by this prior solution with a dialog prompting the user to select file names for the pictures and storage locations.

Brief Summary Text (8):

This prior solution fails to satisfy user expectations regarding the behavior of the pasted pictures because, after the first save within the HTML format, the pasted pictures become

separate linked files. For example, deleting a link does not result in the removal of the linked content from the file system. A change to the linked content in one copy of a document can result in the unintended change of this linked content in other copies of the document. In contrast to a save operation of an electronic document having embedded content, saving a copy of a document with linked content does not result in saving a copy of the linked content. Likewise, saving a document over an existing document does not result in the deletion of linked content in the existing document. Adding new linked content to a document can result in an unexpectedly overwrite of existing content in the document. Also, this prior solution typically handles only embedded images and fails to support other varieties of embedded content, such as embedded stylesheets, embedded web pages, embedded framesets, etc.

Brief Summary Text (10):

A third prior solution operates to save all content in an HTML-formatted document, both linked and embedded, in a special single file containing embedded files. Although this single file solution addresses some of the desired behaviors expected by users of embedded content, this solution also introduces unacceptable limitations because all content in the document is now treated as embedded content, even linked content. In other words, this single file solution satisfies selected user expectations for embedded content but violates all expectations for linked content. In addition, the single file is typically not formatted as an HTML document. This means that the file is not directly readable by browsers or editable by existing web page editors. Moreover, the single file is typically slower to save and slower to load than a similar HTML-formatted file, because of the inherent disadvantage of loading a large single file rather than progressively loading multiple files over a network connection.

Brief Summary Text (11):

In view of the foregoing, there is a need to fulfill users' expectations of how embedded content should work while also using HTML as the file format. The present invention solves this embedded content management problem for HTML-formatted files by placing information in a primary file that provides a cue to an editing program, such as a web page editor, that a particular file associated with that primary file should be treated as either embedded or linked content.

Brief Summary Text (17):

The present invention offers advantages over the prior art for managing embedded content in an HTML-file environment. If the user deletes apparent embedded content in the authoring environment, the corresponding supporting file is also deleted from the storage mechanism. If the user changes apparent content in one copy of a document, this embedded file will not change in other copies of the document because a separate copy of this file, i.e., the corresponding supporting file, is maintained in a known storage location for each document. If the user saves a copy of a document with embedded content, a supporting file corresponding to the embedded file is created and maintained on the storage mechanism for future reference in connection with edit operations of the document copy. If the user saves a document over an existing document, the apparent embedded content of the existing document is cleaned-up by deleting the supporting files corresponding to that embedded content. Adding new embedded content to a document does not result in an overwrite of existing content, either in that document or in any other document. The present invention also can process types of embedded content other than images, while correctly handling linked content and using standard HTML that is readable by browsers and web page editors.

Detailed Description Text (12):

For some types of files, only one instance of the file can exist for a particular primary file. For example, in Microsoft's "POWERPOINT" program, a single GIF file represents the "next slide" button for each slide. For file types that have only one instance for a certain primary file, a unique identifier can be assigned to the corresponding file based on a fixed string of characters. For the referenced example of the GIF file for the "next slide" button in Microsoft's "POWERPOINT" program, the unique identifier "slide_next.gif" can be assigned to this file.

Detailed Description Text (13):

Multiple instances can exist for other types of files. In this case, a unique identifier can be

created by combining a file name defined by a fixed string with a string of numbers, typically starting with the number 1 for the first instance. For example, if a primary file includes three different pictures, then the selected identifiers for these GIF-formatted images can be "image001.gif", "image002.gif", and "image003.gif".

Detailed Description Text (14):

After selecting a unique identifier for a file representing embedded content, a storage configuration can be selected for that file. Two possible configurations include "flat" and "folder" for file layout. In the flat case, a directory-type storage location can be created and identified by prepending the identifier of the primary file plus a "_". A representative example of this flat file format is shown in Table II, for a directory comprising a primary file identified as "Web Page" and a pair of files associated with this primary file, namely, "filelist.xml" and "image001.gif".

Detailed Description Text (15):

For a folder layout, a folder can be created for embedded files by using the name of the primary file plus the localized term for "files". A representative example of this folder format is shown in Table III, for a primary file identified as "Web Page" and a pair of files associated with this primary file, namely, "filelist.xml" and "image001.gif".

Detailed Description Text (16):

For the folder layout operation, each document has its own folder, thereby eliminating the possibility of a conflict of files having the same identifier within the same folder. Moreover, if files having the same string identifier, such as "Web Page," are placed in the same directory, then an extension can be added to the folder name to avoid collisions. For example, if a pair of files have the same string identifier "Web Page," then different extensions can be added to the folder name, such as folders "Web Page.htm" and "Web Page.html."

Detailed Description Text (19):

The file list, which references the primary file, can keep track of the embedded content in this main electronic document. For an exemplary embodiment, the file list is implemented as an XML file called "filelist.xml", and can include <o:File> tags and a <o:MainFile> tag. The <o:MainFile> tag has a single attribute ("o:HRef") that points to the primary file, referenced below as "Web Page." <o:MainFile o:HRef="./Web Page.htm"/>

Detailed Description Text (53):

Turning first to FIG. 6, the process begins at the START step 605 and proceeds to step 610. An inquiry is conducted in step 610 to determine, for the file type logically represented by the support file, whether a single instance of the support file exists for the main electronic document, namely the primary file. If so, the "YES" branch is followed from step 610 to step 615 and a fixed string is assigned as a unique identifier to this support file. The process then proceeds from step 615 to the END step 625.

Detailed Description Text (54):

In contrast, if the response to the inquiry is negative, the "NO" branch is followed from step 610 to step 620. A unique identifier is assigned in step 620 to the support file based on the combination of a fixed string and a unique instance number. For example, the first instance of the support file type can be identified by a fixed string and the instance number 001 (the number 1 preceded by a padding of one or more zeroes). The second instance of this support file type includes the identical fixed string and a different instance number, typically in numerical order, such as 002, and so forth for other instances of this support file type. The process then terminates at the END step 625.

Detailed Description Text (56):

In the event that the response to the inquiry of step 710 is negative, the "NO" branch is followed to step 720, where a folder configuration is selected. In step 725, the folder created in step 720 is assigned a folder identifier based on the identifier for the primary file and a "local term" for files. For example, if the primary file identifier is "web page" for the main document, and the "local term" for files is the term "files", the folder identifier comprises "web page files". The support file is then placed in this identified folder on the storage

mechanism in step 730. The process then terminates at the END step 735.

Detailed Description Text (59):

Those skilled in the art will appreciate that the present invention is not limited to HTML file formats, and that the inventive concepts can be extended to a single-file Web page format, such as MIME HTML.

Detailed Description Paragraph Table (1):

TABLE II Web Page.htm Web Page filelist.xml Web Page image001.gif

Detailed Description Paragraph Table (2):

TABLE III Web Page.htm Web Page files image001.gif filelist.xml

CLAIMS:

4. The computer-readable medium of claim 3, wherein the step of assigning a unique identifier to the support file comprises: for the identified type of content for the embedded file represented by the support file, determining whether a single instance of the support file exists for the primary file; in the event that a single instance of the support file exists for the primary file, then assigning a fixed string as the unique identifier for the support file; otherwise, assigning a combination of a fixed string and a unique instance number as the unique identifier for the support file.

14. The computer-implemented method of claim 13 wherein the step of assigning a unique identifier to the support file comprises: for the identified type of content for the embedded file represented by the support file, determining whether a single instance of the support file exists for the primary file; in the event that a single instance of the support file exists for the primary file, then assigning a fixed string as the unique identifier for the support file; otherwise, assigning a combination of a fixed string and a unique instance number as the unique identifier for the support file.

[Previous Doc](#)

[Next Doc](#)

[Go to Doc#](#)

[First Hit](#) [Fwd Refs](#)[Previous Doc](#)[Next Doc](#)[Go to Doc#](#)

Generate Collection

Print

L13: Entry 9 of 16

File: USPT

Mar 19, 2002

DOCUMENT-IDENTIFIER: US 6360254 B1

TITLE: System and method for providing secure URL-based access to private resources

Brief Summary Text (2):

The present invention relates to electronic commerce and user authentication. More particularly, the invention relates to methods for enabling users of a Web site or other information system to efficiently and securely access private Web pages and other types of restricted resources.

Brief Summary Text (4):

In the field of Internet commerce, it is common for businesses to provide customers restricted access to data, server functionality, and other types of resources via a Web site. For example, a user of an online merchant's Web site may be given restricted access to a database record which contains the user's account information, or to a Web page which allows the user to place an order with the merchant.

Brief Summary Text (8):

The present invention addresses the above and other limitations by providing a system and method in which users access private Web pages, data records and/or other restricted resources using automatically-generated private URLs (Uniform Resource Locators). The private URLs allow authorized users to access the private resources without the need to enter a username, password, or other authentication information, and without the need to download special authentication software or data to the user's computer. The system and method are particularly useful for providing users with secure access to data records and functionality associated with a personal account, but may be used in a wide range of other practical applications.

Brief Summary Text (9):

In accordance the invention, each "private resource" (a resource to which remote access by a particular user or group of users is desired) of a set of private resources is automatically assigned a private URL which includes a fixed character string and a unique token. For example, a private Web page for a particular user may be assigned the private URL

Brief Summary Text (13):

When the user selects the hyperlink or otherwise requests a private URL, a server application running on the Web site determines whether the token is valid. If the token is found to be valid, the server application permits the user to access the resource. Access to the resource may, for example, involve the generation of a private Web page that is transmitted to the user. The private Web page may include information from one or more private database records, and/or may include a confirmation that a particular transaction associated with the private URL was performed. The server application may be configured to invalidate the tokens (and thus the private URLs) after a single use, or after a predetermined period of time.

Brief Summary Text (14):

One benefit of the above-described URL generation/validation method is that it provides a very simple and efficient mechanism for allowing users to access private resources, such as Web pages which contain personal account information. Another benefit is that the private URLs can be generated and issued to a user (and subsequently validated) even if very little information is known about the user. For example, the method can be used where the only information known about the user is the user's email address, as in the email-based recommendation service described below. Different levels of security can therefore be used for different types of

transmissions. Another benefit is that the level of security can be controlled by adjusting the size of the token space, such as by adjusting the number of bits used to represent token values. Another benefit is that the method does not require the user to download any special authentication program, or to consistently use the same computer.

Drawing Description Text (7):

FIGS. 5-7 illustrate a form Web page (FIG. 5), an email document (FIG. 6), and a private Web page (FIG. 7) in an email recommendation service embodiment of the invention.

Drawing Description Text (8):

FIGS 8-10 illustrate a form Web page (FIG. 8), an email document (FIG. 9), and a private Web page (FIG. 10) in an electronic gift certificate embodiment of the invention.

Detailed Description Text (4):

The present invention provides an automated method for enabling remote users of a Web site or other information system to securely access private Web pages or other resources without having to enter a password or other authentication information. The method is preferably implemented in a multi-user system in which different users have access to different private resources. (The term "private" is used herein to indicate a correspondence to a particular user or subset of users of a larger community of users.) The invention also provides several practical applications for the method, the details of which are described below under the heading "Example Applications."

Detailed Description Text (5):

Briefly, the method involves generating a unique, private URL (Uniform Resource Locator) that corresponds to a private resource, and conveying the URL (preferably by email) to the corresponding user or group of users that are to have remote access to the resource. Application code running on the server is used to validate the URLs and to thereby restrict access to the private resources. Each resource may, for example, be in the form of a private Web page which includes data from one or more private data records of a database or files of a file system. The resources may additionally or alternatively comprise server application functionality that allows users to perform particular types of transactions.

Detailed Description Text (11):

When a user attempts to access a resource which is referenced by a URL of the appropriate format (e.g., the format shown above), a token validation program is invoked to validate the token (and thus the URL). There is no need for the user to enter a username, password, or other authentication information during this process. If the token is found to be valid, the user is permitted to access the private resource; otherwise, access to the resource is denied. Access to the resource may, for example, involve the generation of a private Web page that is transmitted to the user. The private Web page may include information from a private database record, and/or may include a confirmation that a particular transaction associated with the URL was performed.

Detailed Description Text (13):

Although email is preferred, other methods may be used to communicate the private URLs to users. For example, a private URL could be communicated as a hyperlink within an ordinary Web page, such as when a user initially sets up an account. Further, the URLs could be communicated by telephone, facsimile, or using another type of Internet document transmission protocol. Further, the token and non-token portions of the URLs could be transmitted to the user separately, such as by transmitting the token by email and the fixed URL portion by a publicly-accessible Web page.

Detailed Description Text (18):

In one embodiment, the issued tokens (and thus the private URLs) automatically expire following a predetermined time period, such as 3 days. This may be accomplished, for example, by encoding time information (such as a creation or expiration time/date) into the tokens, or by storing such time information in a look-up table. The tokens may additionally be caused to expire after a pre-specified number of uses. For example, if the information is sufficiently sensitive, the URL may be set to expire after a single access to the private Web page. Upon the expiration of

a user's private URL, a new URL may be generated automatically and conveyed to the user (such as by email). Alternatively, the URLs may be generated and conveyed to the user when the user performs a particular type of action on the Web site, such as submitting a form. Providing URLs which automatically expire further enhances security by (a) limiting the utility of a valid URL that is discovered by improper means, and (b) in implementations in which replacement URLs are not immediately generated upon expiration, reducing the number of valid tokens.

Detailed Description Text (21):

FIG. 1 illustrates a Web site 30 which is accessible to remote user computers 32 via the Internet, and illustrates a basic set of Web site components that may be used to implement a preferred embodiment of the invention. In the example shown, the private resources include private data records 46 that are accessed using private Web pages. For purposes of illustration, it may be assumed that each data record 46 corresponds to respective user, and is accessible via a respective private Web page which has a private URL assigned thereto. In the illustrated embodiment, the private URLs are conveyed to the users by email within hyperlinks. (The term "hyperlink" is used herein to refer collectively to a URL and to the corresponding document element which is selectable by a user to access a resource associated with the URL.)

Detailed Description Text (22):

As depicted by FIG. 1, the Web site 30 includes a Web server 36 which provides access to a store 38 of HTML (Hypertext Markup Language) content. The Web server may, for example, include a commercially available Web server program which runs on one or more general-purpose Unix or Windows NT based computers. The HTML store 38 stores various Web page components, including HTML that are used to dynamically generate private Web pages.

Detailed Description Text (24):

The back-end database 42 includes private data records 46, wherein different records are private to different users (or possibly groups of users). The data contained in the data records 46 is dependent upon the type of service provided by the server application 40. Each data record may, for example, include account information for a respective user of the Web site, and/or may include data that is dynamically updated by other processes of the Web site. The data records 46 are retrievable from the database 42 using email addresses or other identifiers of corresponding users. The private data records 46 are accessible to users via private Web pages generated by the Web server 36.

Detailed Description Text (29):

As depicted by event A, the server application 40 generates and transmits to the user 70 an email document 72 which includes a hyperlink 74 to a private URL. The email document 72 may also include a description of the function(s) performed by the hyperlink 74. To generate the hyperlink 74, the server application 40 initially invokes its token generation code 50 (FIG. 1) to generate a token that uniquely corresponds to the user's private data record 46. This token is then combined with a fixed character string to form the URL, and the URL is incorporated into the email document 72 within the HREF (hypertext reference) portion of the hyperlink 74. As is conventional, the URL itself may be hidden from the user during normal display of the document, and may be represented within the document as underlined text or a button. Alternatively, the URL may be made viewable so that the user can easily copy the URL for future use. As indicated above, the private URL could alternatively be communicated to the user using an ordinary Web page or another transmission method.

Detailed Description Text (32):

As depicted by event C, if the token (and thus the URL) is found to be valid, the Web server 36 generates and returns a private Web page 78 which contains data from the private record 46. The private Web page may, for example, include a form for allowing the user to update the information. The private Web page may additionally or alternatively include a message confining that a particular action associated with the private URL was performed. For example, the private Web page 78 could confirm that an electronic gift certificate amount was credited to the user's account (as in FIG. 10, described below).

Detailed Description Text (37):

To generate a token (FIG. 3A), the user's email address is initially converted into a unique,

36-bit email ID (step 130). Any algorithm which reversibly converts alphanumeric strings into fixed-length binary values can be used for this purpose. In step 132, the email ID and a time stamp which represents the current date and time are combined into a 64-bit integer value. The time stamp may, for example, be a Unix-based time stamp (or a rounded-off version thereof) which represents the number of seconds since 1970. Inclusion of the time stamp provides a mechanism for later determining whether the token has expired. The time stamp may alternatively be in the form of an expiration date/time.

Detailed Description Text (41):

If the email ID and the time stamp are valid, the email ID is used to retrieve the private record 46 to be accessed (step 158). The private record 46 may, for example, be a record of the user's account data. This record, or a portion of the record, may be returned to the user's browser 66 within a private Web page. As described above, the server application 40 may additionally or alternatively update a database or perform some other type of action in this event.

Detailed Description Text (49):

FIGS. 5-7 illustrate, in example form, a particular application in which the server application 40 (FIG. 1) implements an automated email-based recommendation and notification service of the Amazon.com Web site. As depicted by the Web page of FIG. 5, the user subscribes to the service by completing and submitting an online form. The form prompts the user to enter the user's Internet email address, and to select one or more categories of books, music, gifts and videos (by selecting appropriate check boxes) for which to receive information about Amazon.com's product offerings. In the preferred embodiment, the user can, but need not, have a preexisting account with the Amazon.com Web site. Users can subscribe to the service without submitting or otherwise disclosing any additional information.

Detailed Description Text (51):

The email document 72 also includes a hyperlink 74 to a private Web page 78 (FIG. 7) that can be used to modify the subscription profile or cancel the service. The URL referenced by the hyperlink 74 is generated and validated as described above. The user can thus subscribe to the service, and thereafter securely modify the subscription profile, without the need for a password, username, or other authentication information, and without the need to disclose any personal information other than an email address and subscription profile. With reference to FIG. 7, the URLs (not shown) associated with the "Submit Changes" and "Unsubscribe" buttons may include the token, a session ID, or some other identifier that can be used to track the user.

Detailed Description Text (54):

As shown in FIG. 10, selection of the hyperlink 74 also causes the Web site to return a private gift certificate redemption page 78. The private redemption page 78 includes a message 284 confirming that the gift certificate amount was credited to the recipient's account. From this Web page 78, the user can initiate a search of the merchant's offerings. Additional details of the illustrated gift certificate system are described in U.S. Pat. application No. 09/153,632, filed Sep. 15, 1998, the disclosure of which is hereby incorporated by reference.

Detailed Description Text (55):

Another practical application (not separately illustrated) involves sending the user 70 (FIG. 2) an email document 72 or a Web page which includes a one-time-use URL (preferably as a hyperlink 74) to a private discount page 78. The discount page 78 may, for example, give the user a 10% discount off the user's next purchase. Other users of the system would be sent like emails but which contain different tokens. In this application, the server application 40 would use the tokens to prevent users from obtaining multiple discounts. This may be accomplished, for example, by deleting each issued token from a table once the token has been used.

Detailed Description Text (56):

Another application (not separately illustrated) involves providing restricted access to account information in the context of orders for goods. In one embodiment, for example, when a user 70 (FIG. 2) places an order on a merchant's Web site 30, the private URL/hyperlink 74 is transmitted to the user via an order confirmation email 72, and provides access to a private Web page 78 that includes order status information or other account-related information. The

private Web page 78 may include links or a form for allowing the user to modify the order.

Detailed Description Text (57):

In another practical application (not separately illustrated), the user 70 is a supplier of goods that are sold via the Web site 30 of a retailer, and the server application 40 is used to place orders with the supplier. To place an order, the server application 40 generates and transmits to the supplier an email document 72 which preferably includes the following: (a) a description of the goods being ordered, (b) a hyperlink 74 to a private URL (generated as described above), and (c) a message instructing the recipient to select the hyperlink to confirm the order. The email document 72 may be generated automatically, such as when inventory falls below a certain threshold, or may be generated in response to an action performed by a user. When the user 70 selects the hyperlink 74, the server application 40 updates a database to indicate that the order was confirmed, and the Web server 36 returns a private Web page 78 which includes information (e.g., order history, balance due) about the supplier's account with the retailer.

Detailed Description Text (59):

Another application (not separately illustrated) involves providing security in an electronic greeting card system, such as the Web-based system of Blue Mountain Arts. In this type of system, the private URLs reference electronic greeting card Web pages 78 that are customized by other users. For example, by accessing a greeting card creation area of the Web site, a user can select and customize a private Web page/greeting card and have the URL of this card sent by email to a designated recipient 70. Use of the URL generation/validation methods of the invention in this environment would provide increased privacy of the greeting cards.

Detailed Description Text (60):

Although this invention has been described in terms of certain preferred embodiments and applications, other embodiments and applications that are apparent to those of ordinary skill in the art, including embodiments which do not provide all of the features and advantages set forth herein, are also within the scope of this invention. For instance, although the invention has been described in the context of access to Web pages and other Internet resources, the underlying methods can also be used to provide secure access to other types of addressable resources, including resources accessed through proprietary protocols. Accordingly, the scope of the present invention is intended to be defined only by reference to the appended claims.

CLAIMS:

2. The method of claim 1, wherein the resource comprises personal account information stored in a database, and accessing the resource comprises returning to the user a private Web page which includes the account information.

3. The method of claim 2, wherein the account information includes a subscription profile for an email-based subscription service, and the private Web page includes an electronic form for modifying the subscription profile.

4. The method of claim 1, wherein accessing the resource comprises transmitting to the user a Web page that contains an electronic greeting card created for the user by another user.

23. The computer system of claim 16, wherein the server system responds to a private URL that includes a valid token by generating and returning a private Web page.

24. The computer system of claim 23, wherein the server system implements an email-based subscription service in which the tokens are used to provide secure access to private Web pages that allow users to modify personal subscription profiles.

27. The computer system of claim 16, wherein the server system implements an electronic greeting card system in which the tokens are used to provide secure access to private greeting card Web pages.

30. In a Web site system of a merchant, a computer-implemented method of providing customized

information to a user about products and/or services available from the merchant, comprising:

obtaining an email address and a subscription profile from the user, the subscription profile indicating product and/or service categories selected by the user;

transmitting to the user at least one email document which contains descriptions of products and/or services, the descriptions selected based on the subscription profile;

generating and transmitting to the user a private uniform resource locator (URL) which provides access to a private Web page for at least securely revising the subscription profile, the URL containing a token which is generated using a method which distributes tokens substantially randomly over a token space; and

responding to a client request for the private URL by returning the private Web page without requiring entry of authentication information.

[Previous Doc](#)

[Next Doc](#)

[Go to Doc#](#)